Abstract
Presentation
Back to Agenda
Next Abstract

Beamforming for Radar Systems on COTS Heterogeneous Computing Platforms

Mr. Jeffrey Rudin
Mercury Computer Systems, Inc.
Phone: (978) 967-1686
Fax: (978) 256-8596
Email: jrudin@mc.com

Abstract:

The introduction of high-speed analog-to-digital converters has resulted in many of the traditional front-end and sub-array combining functions of multi-function, phased-array radar systems being performed in the digital rather than in the analog domain. Due to the intense amount of processing that is required, many of these functions had to be realized in hardware. This was originally accomplished using VLSI ASICs. However, the advent of multi-million gate field-programmable gate array (FPGA) has permitted these complex digital processing functions to be put in small packages with a degree of design flexibility that is normally associated only with software. This permits more of the radar functions to be realized in commercial off-the-shelf (COTS) hardware by obviating the need of full-custom VLSI in many cases.

The incorporation of FPGA technology into COTS processing subsystems permits more complex designs to be created than could be achieved by general-purpose or digital signal processors alone. Simply incorporating FPGAs into single board computers could solve many signal processing problems. However, because of the complexity of the signal processing in a multi-function radar system, a distributed, parallel-processing architecture is usually required. In addition, the trend toward an increasing number of input channels and the formation of a greater number of simultaneous beams requires a high degree of interconnection among the processing elements. Therefore, the technology used to interconnect the computing elements must be flexible enough to accommodate different architectures and system requirements. Furthermore, the interconnection technology should be scalable enough to enable early design prototyping as well as system deployment over a wide range of mission platforms.

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **20 AUG 2004**	2. REPORT TYPE **N/A**	3. DATES COVERED **-**

4. TITLE AND SUBTITLE **Beamforming for Radar Systems on COTS Heterogeneous Computing Platforms**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Mercury Computer Systems, Inc.**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S)
	11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

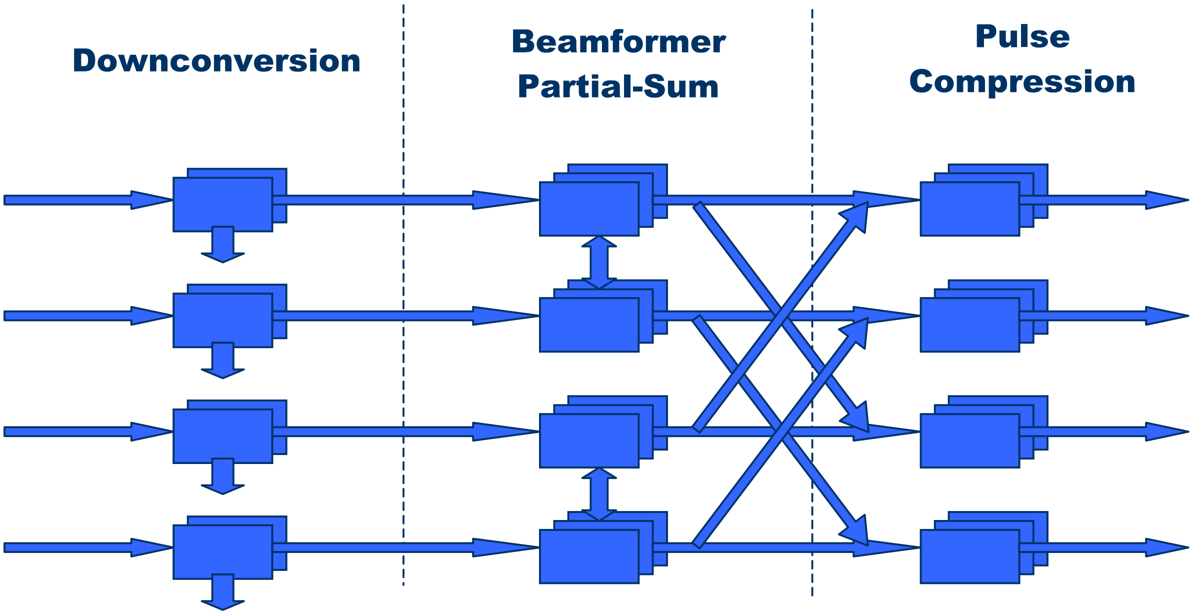
13. SUPPLEMENTARY NOTES
See also ADM001694, HPEC-6-Vol 1 ESC-TR-2003-081; High Performance Embedded Computing (HPEC) Workshop (7th)., The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**UU**	**32**	

Abstract
Presentation
Back to Agenda
Next Abstract



Example Interconnection of Radar Front-End Processing

This paper focuses on the impact of using a heterogeneous distributed computing system for digital beamforming in a multi-function radar system. The interconnection of FPGAs requires balancing the utilization of FPGA resources for endpoint logic I/O with that for processing requirements. A balance must also be struck in the mapping of functions between the FPGAs and the programmable processors in a heterogeneous system. Very frequently, the scaling of particular requirements will require the interconnection topology to change rather than just scale. We examine several different sets of requirements and the subsequent mapping to the heterogeneous computing platforms and the tradeoffs involved. Particular focus is given to the changes in functional allocation and the resulting system topologies.

Computer Systems, Inc. MERCURY

Beamforming for Radar Systems on COTS Heterogeneous Computing Platforms

Jeffrey A. Rudin

Mercury Computer Systems, Inc.

High Performance Embedded Computing (HPEC) Conference

September 23, 2003

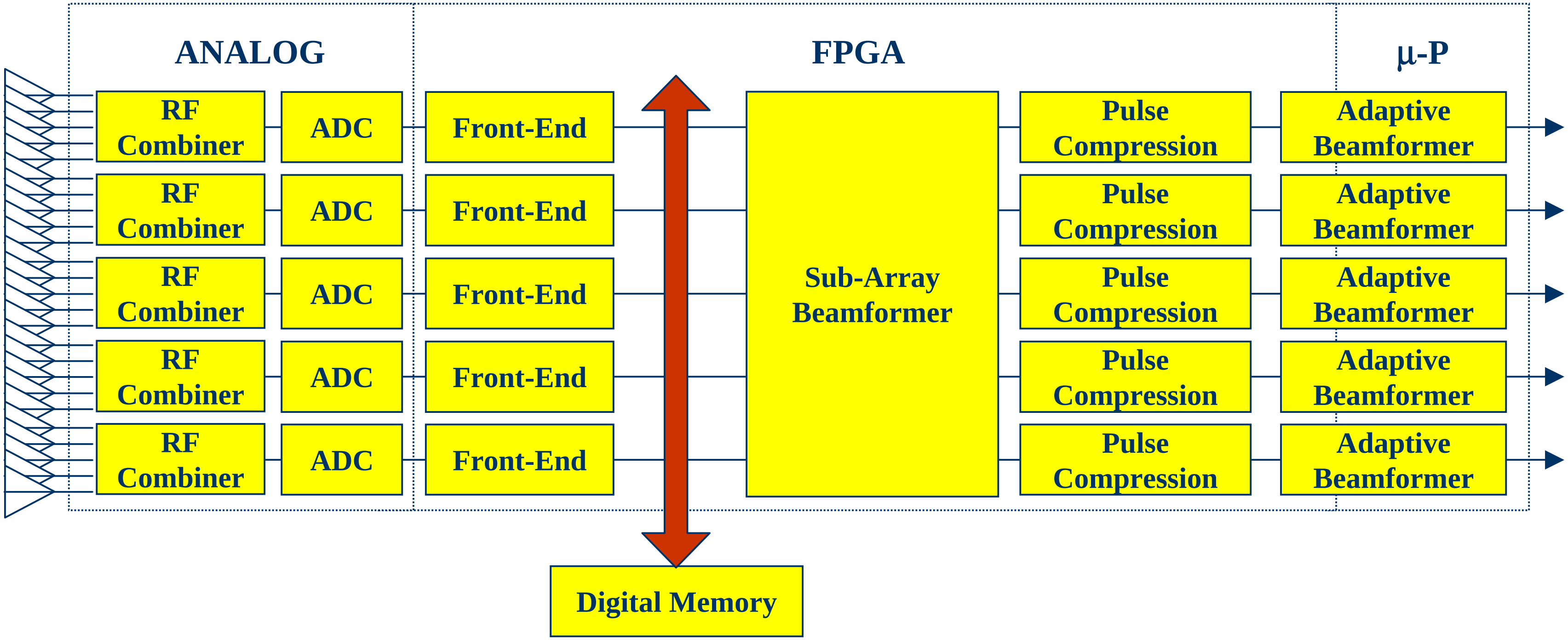
The Ultimate Performance Machine

Outline

- Beamforming Radar System Architecture
- Processing Resources
- Strawman System Analysis
  - Front-End Processing
  - Back-End Processing
  - Beamformer Architectures
- Summary

Radar System Architecture

- Beamforming requires massive dataflow and computation
  - ADC precision and data rate are chosen to provide high dynamic range and and wide signal bandwidth
  - High number of input channels required in modern phased array radars to produce multiple beams and nulls

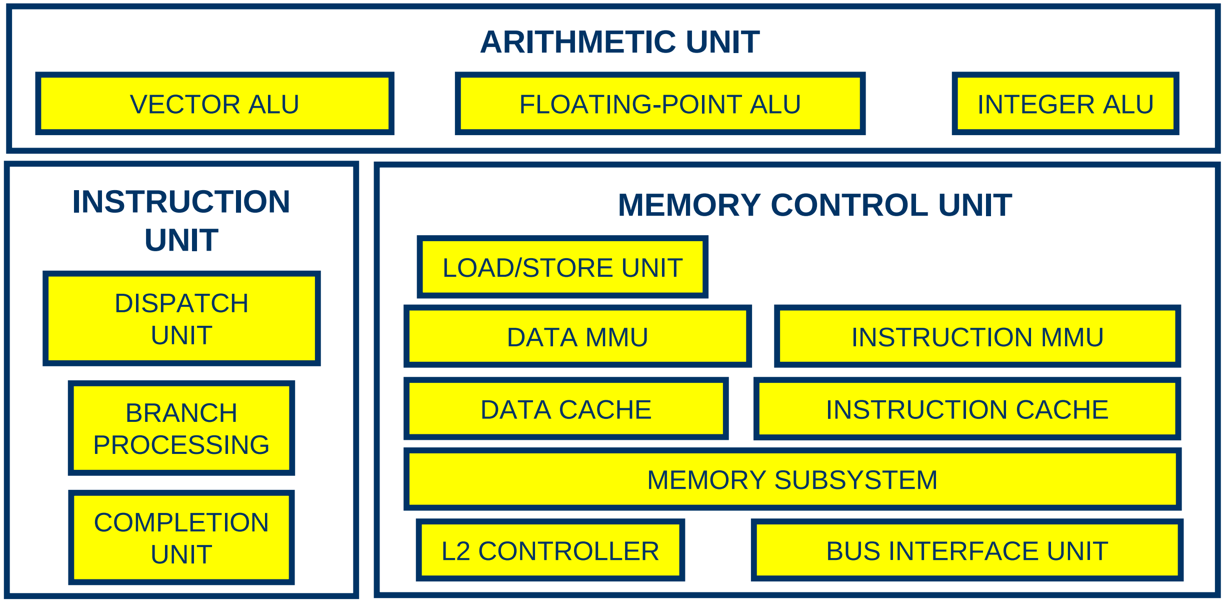


Processing Resources

- Microprocessors
  - Fixed processing, I/O, and memory architecture
  - Task context switch requires microseconds
  - Native floating-point available
  - Low interaction between code modules
- FPGAs
  - Customizable processing, I/O, and memory architecture
  - Task context switch requires reconfiguration -- milliseconds
  - Floating-point must be built or bought
  - Considerable interaction between IP cores
  - Signal propagation issues
  - Currently harder to program than microprocessors

PowerPC Microprocessor

- 400 - 1000 MHz clock speeds
- 133 MHz system bus (MPC74xx) -- 851 MB/s
- 64-bit integer and floating-point units
- 128-bit AltiVec vector processing unit
- Pipelined instruction unit
- 32 kB instruction and data caches
- Up to 2 MB L2 cache



ARITHMETIC UNIT

VECTOR ALU

FLOATING-POINT ALU

INTEGER ALU

INSTRUCTION UNIT

DISPATCH UNIT

BRANCH PROCESSING

COMPLETION UNIT

MEMORY CONTROL UNIT

LOAD/STORE UNIT

DATA MMU

INSTRUCTION MMU

DATA CACHE

INSTRUCTION CACHE

MEMORY SUBSYSTEM

L2 CONTROLLER

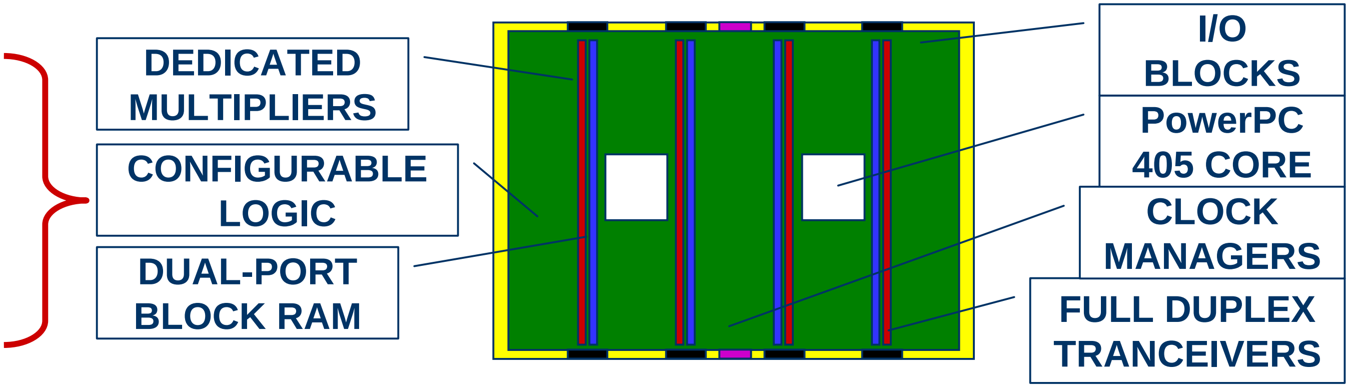
BUS INTERFACE UNIT

Virtex-II Pro FPGA

- Clock speeds lower than processors: 100 - 200 MHz clocks
- Up to 20 full-duplex multi-gigabit transceivers.
- Many DSP supporting features

Device	Gigibit Tx/Rx	Logic Slices	18-Bit Multiplier	18K-Bit Block RAM	Clock Manager	I/O Pads	CPU Blocks
XC2VP40	12	19,392	192	192	8	804	2
XC2VP50	16	23,616	232	232	8	852	2
XC2VP70	20	33,088	328	328	8	996	2
XC2VP100	20	44,096	444	444	12	1,164	2

REGISTERS
LUT'S
CARRY LOGIC
MULTIPLEXERS
DISTRIBUTED RAM
SHIFT REGISTERS



DEDICATED MULTIPLIERS

CONFIGURABLE LOGIC

DUAL-PORT BLOCK RAM

I/O BLOCKS

PowerPC 405 CORE

CLOCK MANAGERS

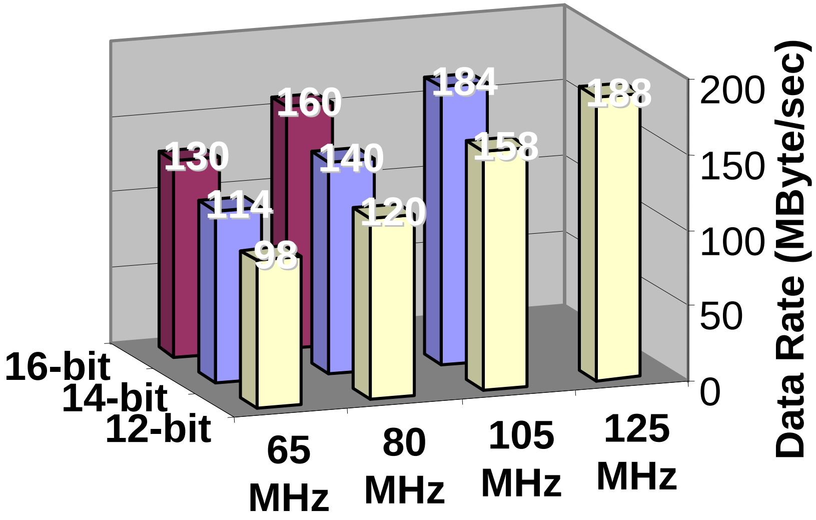
FULL DUPLEX TRANCEIVERS

Each block RAM contains two banks with independent sets of address and data lines
Gigabit transceivers provide over 240 MBps each direction -- over 4800 MBps throughput!

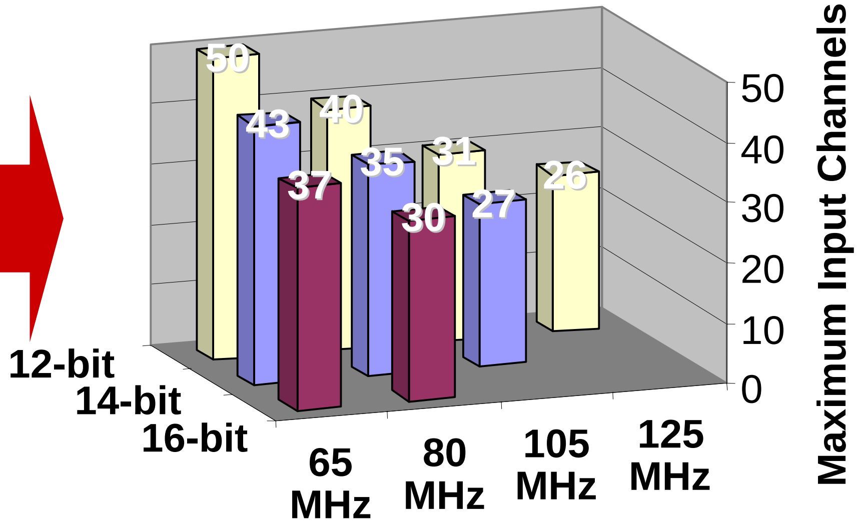
Strawman System Requirements

- Lots of channels -- 80+ input channels
- ADC with “good” bandwidth and dynamic range
    - 100 MSps -- 1.56 - 25 MHz bandwidth using $f_s/4$ sampling
    - 14-bit precision -- over 80 dB dynamic range
- Reasonable implementation risk -- 100 MHz clock

ANALOG/DIGITAL CONVERTER DATA RATES



INPUT CHANNELS PER FPGA USING GIGABIT Tx/Rx



ADC precision and rate and number of channels drive downstream requirements

Front-End Processing

- **Digital Down Converter**
  - fs/4 IF & BW
  - 4x decimation

  } Eliminates the need for numerically controlled oscillators (NCO)
  - 31-tap complex FIR, real symmetric coefficients
  - Usually no bit growth

$$G_{BIT} = \frac{1}{2}\log_2\left(\frac{SNR_o}{SNR_i}\right)$$

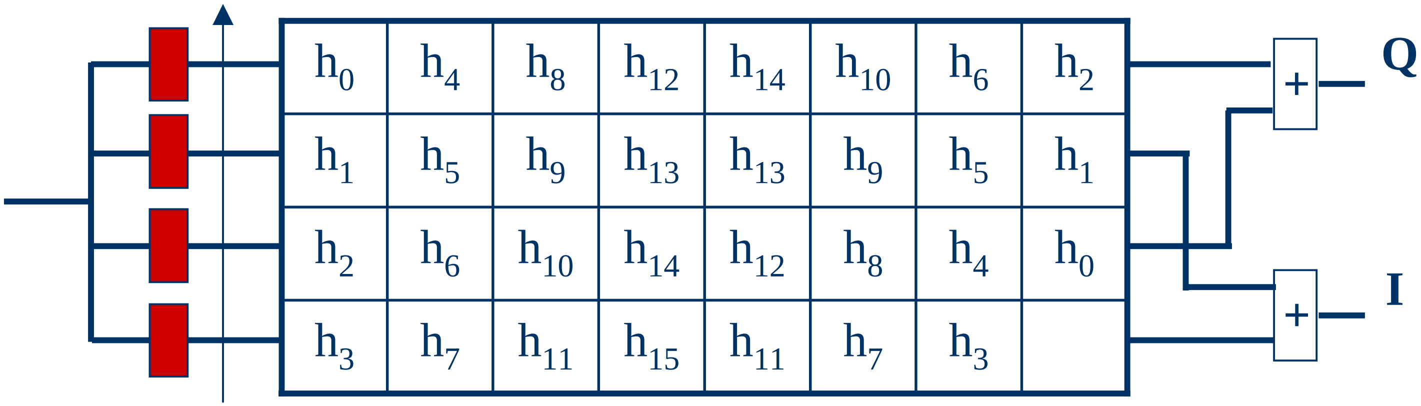
- **Lowpass Decimation Filter**
  - 1x (bypass), 2x, 4x, 8x, and 16x decimation rates
  - 0, 16, 32, 64, 128 taps
  - Real coefficients
  - 0 to 2 bits of bit growth
- **Equalizer**
  - 16-tap, complex coefficients -- cannot generally exploit symmetry
  - Usually no bit growth

Digital Down Converter

- Reduce complexity -- exploit fs/4 center frequency and bandwidth
  - Complex mixing reduces to polyphase commutation
    - Cosine and sine select even and odd samples respectively
      - cos(jnπ/4) = 1, 0, -1, 0, 1,…; sin(jnπ/4) = 0, j, 0, -j, 0,…
  - Exploit polyphase structure for decimation

$$y[n]=\left(\sum_{i=0}^{N-1}h_3[i]x[n-4i]+\sum_{i=0}^{N-1}h_1[i]x[n-4i+2]\right)+j\left(\sum_{i=0}^{N-1}h_2[i]x[n-4i+1]+\sum_{i=0}^{N-1}h_0[i]x[n-4i+3]\right)$$

POLYPHASE fs/4 DDC



h_0	h_4	h_8	h_{12}	h_{14}	h_{10}	h_6	h_2
h_1	h_5	h_9	h_{13}	h_{13}	h_9	h_5	h_1
h_2	h_6	h_{10}	h_{14}	h_{12}	h_8	h_4	h_0
h_3	h_7	h_{11}	h_{15}	h_{11}	h_7	h_3	

Odd number of taps creates symmetries in the FIR coefficients

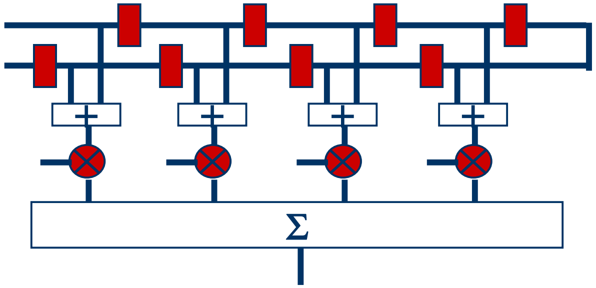
Digital Down Converter

- Reduce complexity -- exploit filter symmetries

Exploit symmetric filter structures for in-phase signal

$$y[n]=\sum_{i=0}^{N/2-1}h[i]x[n-i]+\sum_{i=N/2}^{N-1}h[N-1-i]x[n-i]$$

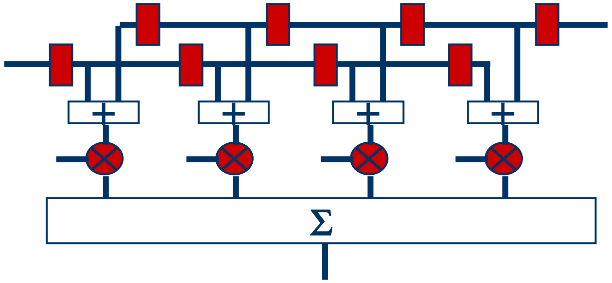
$$=\sum_{i=0}^{N/2-1}h[i](x[n-i]+x[n-({}^{N}\!/_{2}-1-i)])$$



Exploit symmetry pair filters for quadrature signal

$$y[n]=\sum_{i=0}^{N-1}h[i]x_1[n-i]+\sum_{i=0}^{N-1}h[N-1-i]x_2[n-i]$$

$$=\sum_{i=0}^{N-1}h[i](x_1[n-i]+x_2[n-(N-1-i)])$$

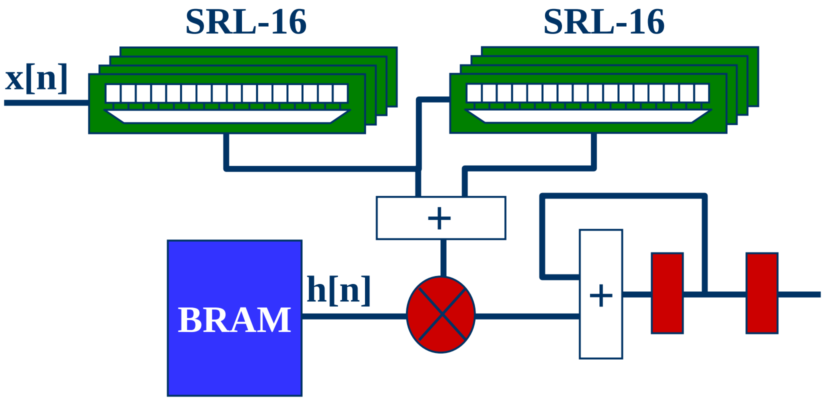


Each tap calculation involves one coefficient and two samples

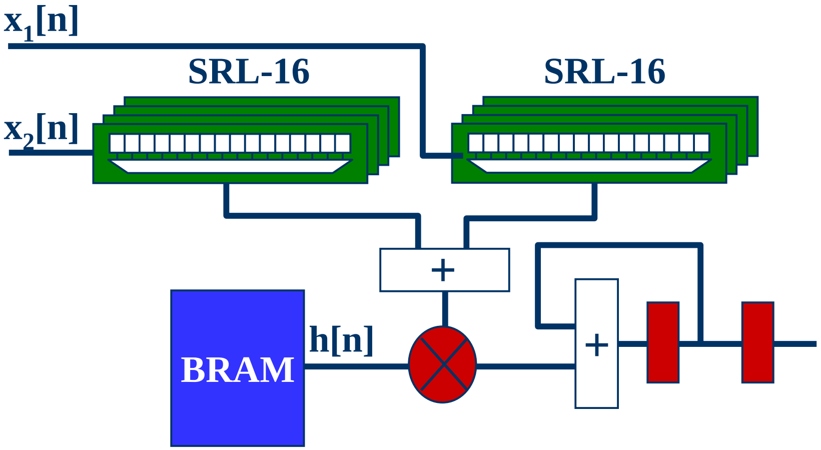
Digital Down Converter

- Reduce complexity -- exploit 4x decimation
  - Use MAC-Engine to do 4 multiplies per input sample
    - Use fclk = 4 x fs to time share multipliers
  - Configure logic slices as shift registers (SRL's) to save BRAM
    - Need to store 3 sets of numbers -- need 2 BRAM's
      - Save BRAM by using logic slices to store both sets of samples

Symmetric Filter



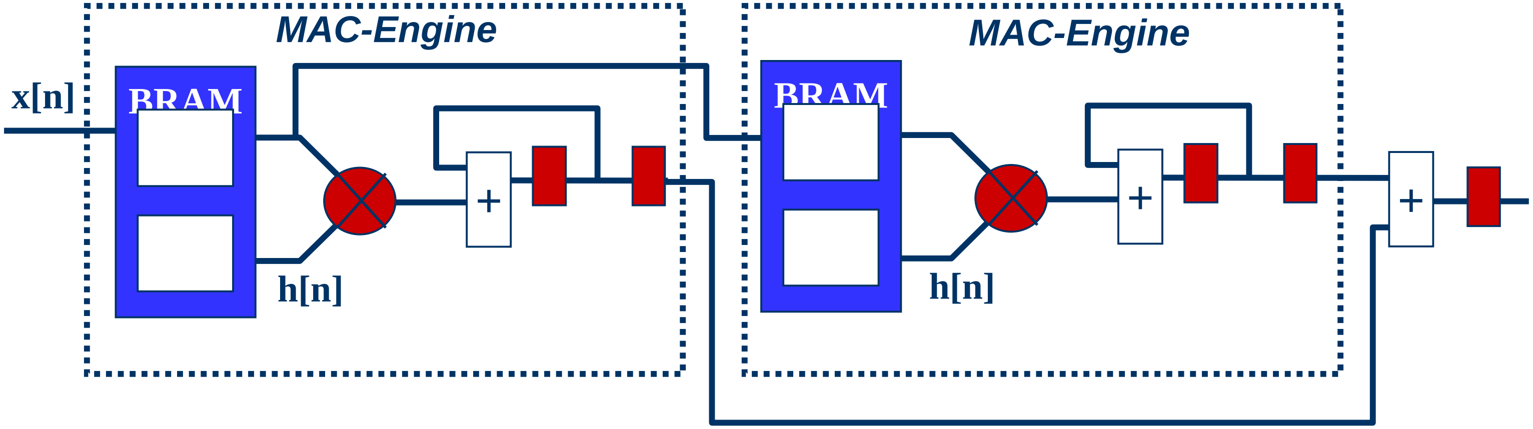
Symmetry Filter Pair



Low Pass Filter

- Reduce complexity -- use MAC-Engine FIR implementation
  - Run multipliers at 4x sample rate -- time share multipliers
  - Exploit constant length-decimation product
    - Single structure handles multiple filter implementations
    - Single clock frequency
  - Use dual-bank feature of BRAM
    - First bank stores samples
    - Second bank stores FIR coefficients

$$N_{MAC-Eng} = N_{Taps} \frac{f_{out}}{f_{clk}}$$

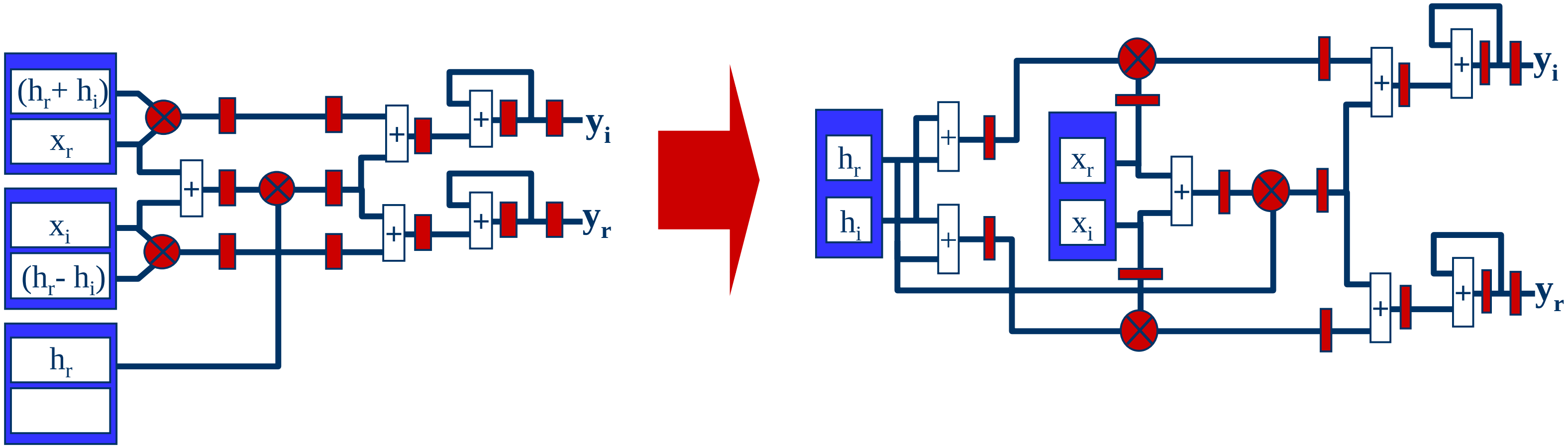


Equalizer

- **Reduce complexity -- reduce number of multipliers and BRAM’s**
  - **Exploit f_{clk}/f_s -- use MAC-Engine**
  - **Implement complex multiply using only 3 MAC-Engines**
    - **Use common product term in complex multiply**

$$\begin{aligned} y_r &= x_r h_r - x_i h_i \\ &= (x_r - x_i)h_r + x_i(h_r - h_i) \end{aligned} \qquad \begin{aligned} y_i &= x_r h_i + x_i h_r \\ &= x_r(h_i + h_r) - (x_r - x_i)h_r \end{aligned}$$

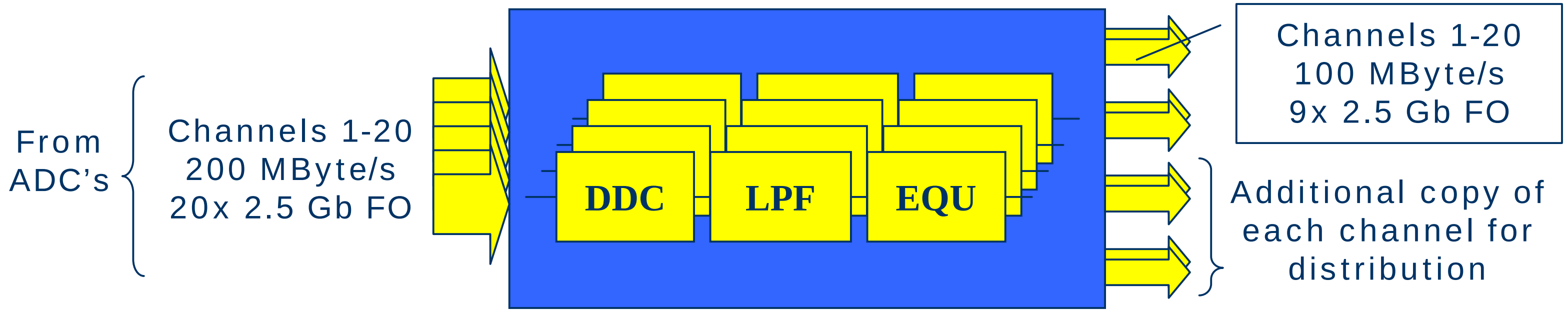
Trade logic slices for multipliers ***Trade logic slices for block RAM***



Front-End Realization

- **FPGA features can be exploited to maximize utilization**
  - **Up to 20 100-MSps channels per FPGA**
  - **DDC with 31-Tap FIR using *only* 3 multipliers/channel**
  - **LPF 16-128 Tap decimating FIR using *only* 4 multipliers/channel**
  - **EQU 16-Tap complex FIR using *only* 12 multipliers/channel**

Digital Receiver Module for 20x 100 MSps Channels on Virtex-II Pro 100



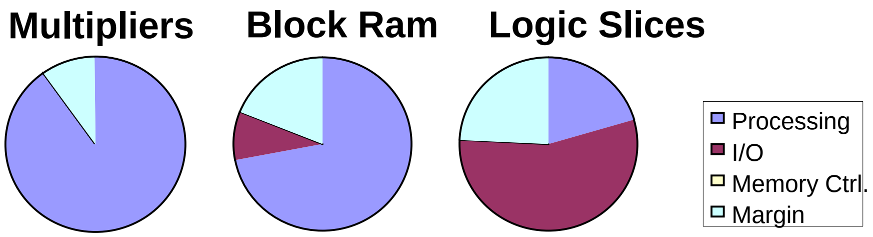
From ADC's

Channels 1-20
200 MByte/s
20x 2.5 Gb FO

DDC LPF EQU

Channels 1-20
100 MByte/s
9x 2.5 Gb FO

Additional copy of each channel for distribution



HIGH FPGA UTILIZATION

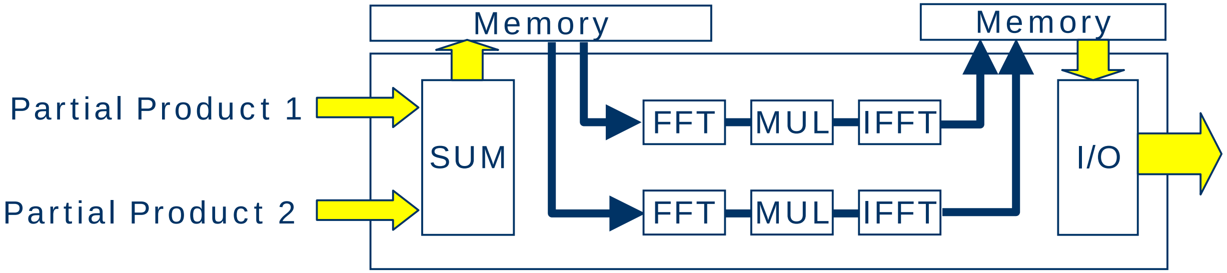
FPGA Utilization for 20x 100 MSps Channels

Back-End Processing

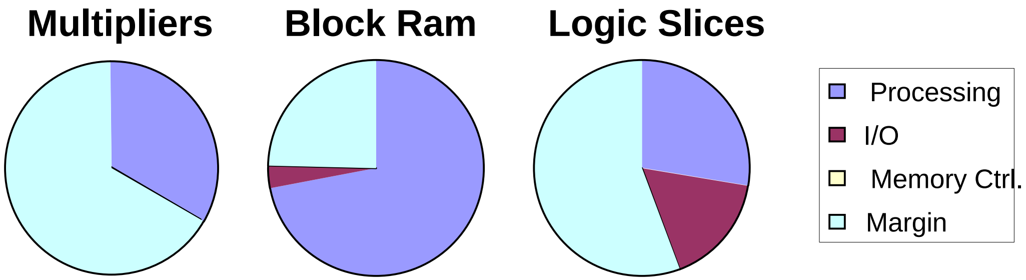
- FPGAs can be used to address data flow requirements that persist in the system until application of adaptive beamforming weights
  - Digital Pulse Compression
    - Fast convolution with FFT IP cores
  - Doppler Processing
    - FPGA FFT IP cores available
  - Adaptive Beamforming Weight Application
    - Similar advantages to those in sub-array beamformer
- FPGAs can augment weight computation
  - QR Decomposition
    - New FPGA solutions may replace microprocessors
  - Cholesky Decomposition
    - Possibly form covariance matrix in adjunct FPGA

Digital Pulse Compression

- FFT IP cores can be used to implement pulse compression
  - 8192-tap FFT @ 25 MSps/channel
  - 6 sub-array channels / FPGA
  - 3-stage pipelined convolver -- 2 convolvers / FPGA
  - Enough resources to sum partial products from beamformer



Doppler processing can be implemented using similar FFT cores



GOOD FPGA UTILIZATION

DIGITAL PULSE COMPRESSION FPGA UTILIZATION

FFT cores tend to be BRAM hungry.

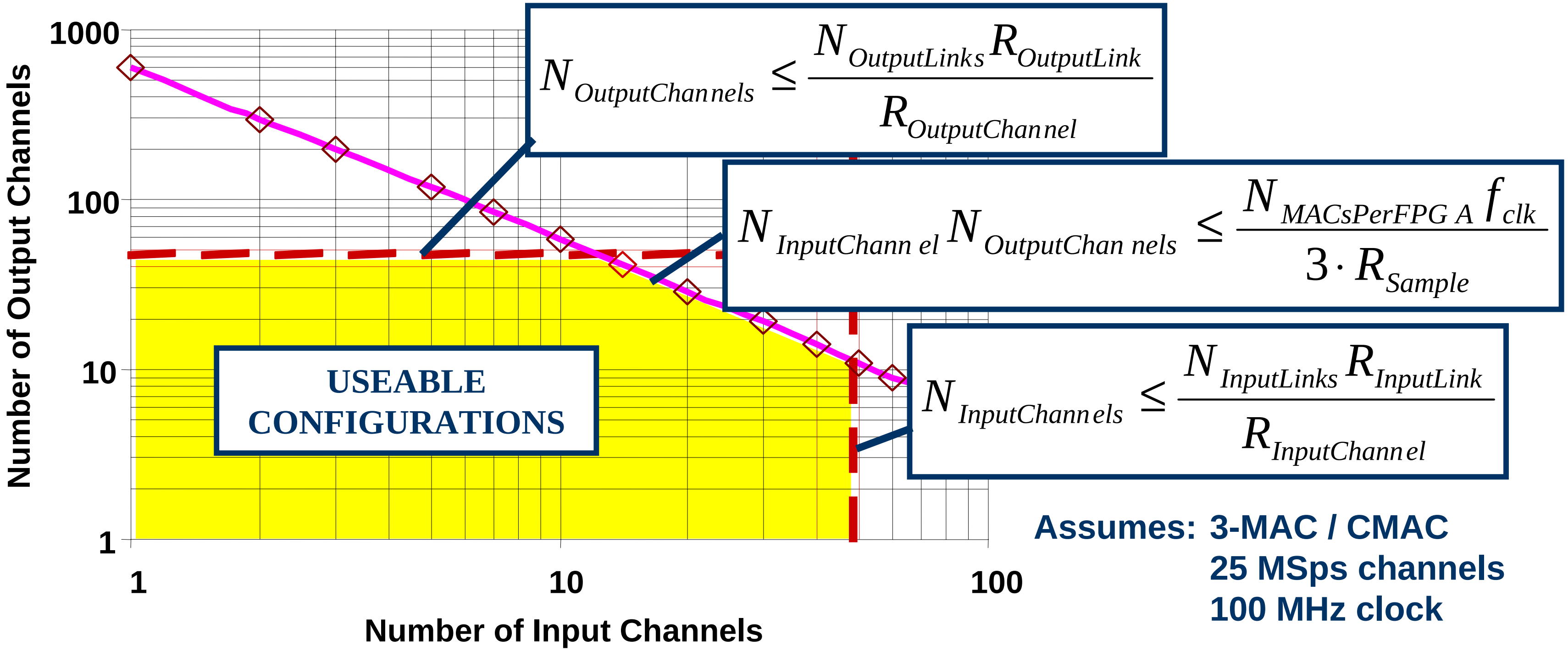
Beamformer Architectures

- Unconstrained Linear Architecture
  - All input channels contribute to each output
- Constrained Linear Architecture
  - A subset of input channels contributes to any output
- Mesh Architecture
  - All input channels contribute to each output

Beamformer Module Constraints

- Basic limits are imposed by I/O and number of multipliers
- Inputs over 18-bits can increase the number of multipliers
  - Keep watch on bit growth in front-end processing

I/O and Multiplier Constraints for Virtex-II Pro 100



$$N_{OutputChannels} \le \frac{N_{OutputLinks} R_{OutputLink}}{R_{OutputChannel}}$$

$$N_{InputChannel} N_{OutputChannels} \le \frac{N_{MACsPerFPGA} f_{clk}}{3 \cdot R_{Sample}}$$

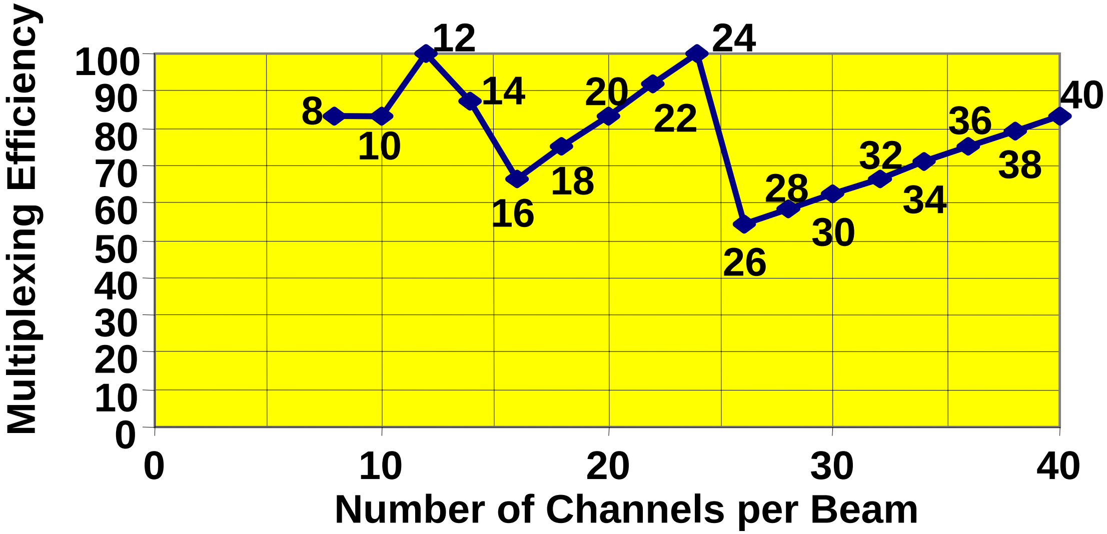
$$N_{InputChannels} \le \frac{N_{InputLinks} R_{InputLink}}{R_{InputChannel}}$$

Assumes: 3-MAC / CMAC
25 MSps channels
100 MHz clock

Beamformer Module Constraints

- Multiplexing must be designed to maximize communication
  - Beam Partitioned output multiplexing may reduce efficiency
  - Alternate multiplexing methods may be necessary

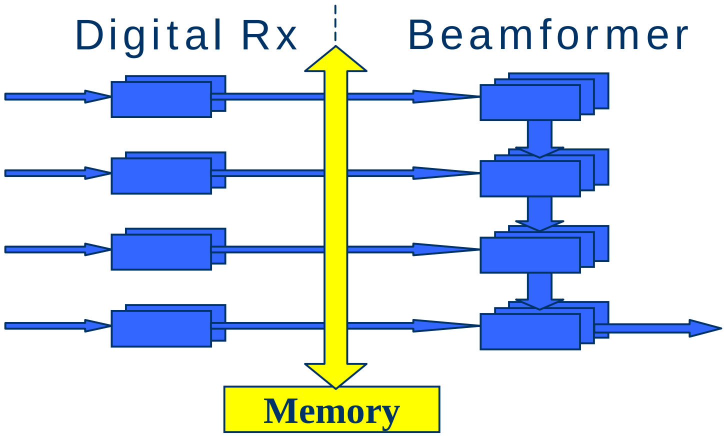
$$\eta_{MUX} = \left(1 - \frac{1}{N_{Links} R_{Link}} \left\lfloor N_{Links} \Big/ \left\lceil \frac{N_{ChannelsPe\quad rBeam} R_{OutputChan\quad nel}}{R_{Link}} \right\rceil \right\rfloor \right)$$



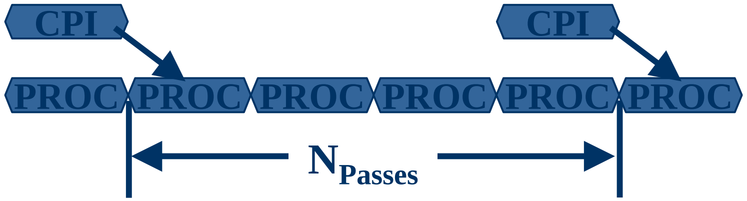
Data can also be partitioned by link: each link carried an integral number of channels

Unconstrained Linear Architecture

- Full MxN unconstrained complex matrix multiply
- Outputs only from a single module
- Processing throughput limited by beamformer module I/O
- Communication latency across beamformer is an issue
- Additional beams can be produced by multiple passes on data
  - Decreases overall radar duty cycle
  - Memory should be located in digital beamformer to save I/O bandwidth
  - Increased beamformer processing speed may be required

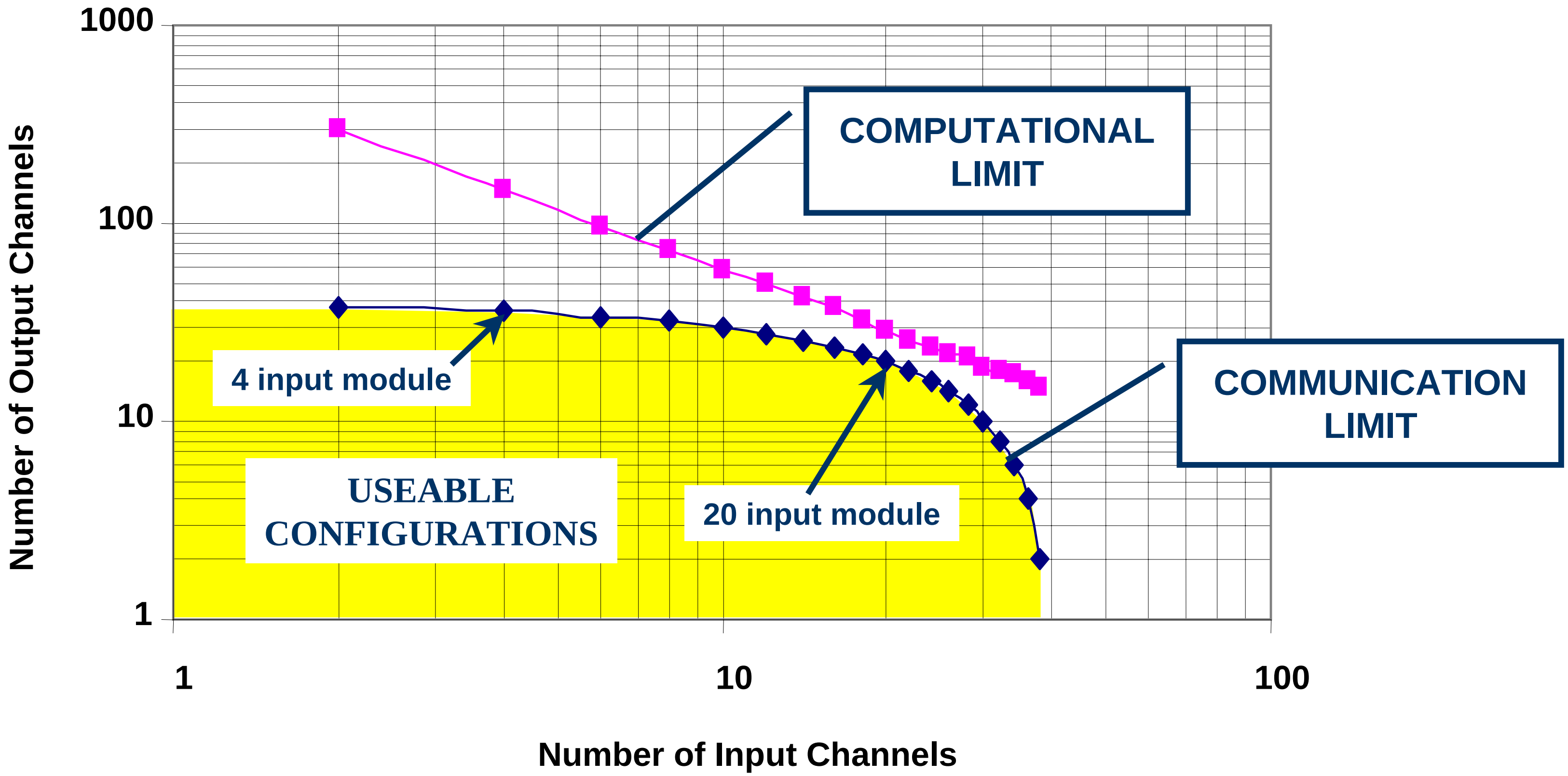


$$\text{Beams}\left\{\begin{bmatrix} Y_1 \\ \vdots \\ Y_M \end{bmatrix}\right. = \begin{bmatrix} H_{11} & H_{12} & \cdots & H_{1(N-1)} & H_{1N} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ H_{M1} & H_{M2} & \cdots & H_{M(N-1)} & H_{MN} \end{bmatrix} \left.\begin{bmatrix} X_1 \\ \vdots \\ \vdots \\ \vdots \\ X_N \end{bmatrix}\right\}\begin{matrix}\text{Input}\\ \text{Sets}\end{matrix}$$



Unconstrained Linear Architecture

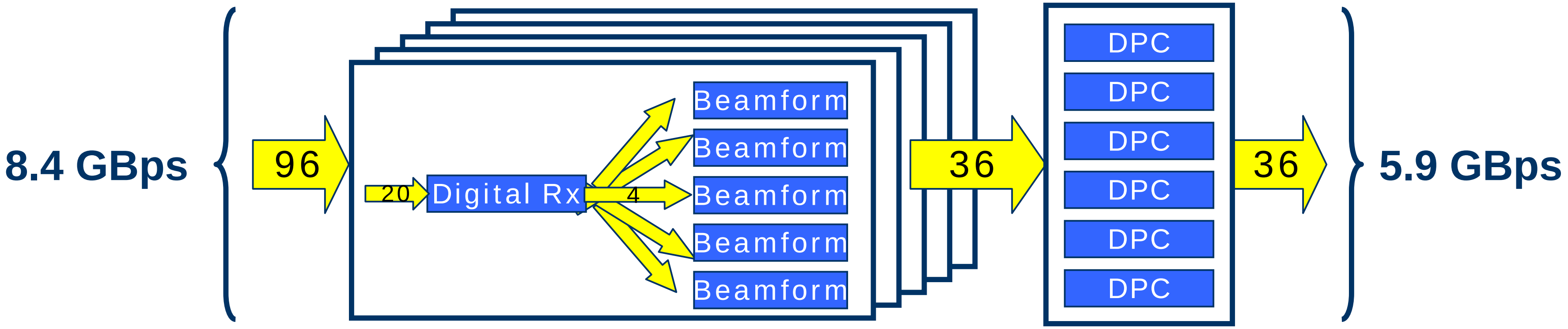
- Unconstrained linear beamformer module is I/O bound
  - Total number of input links plus output links is constant
  - Choice of input to output balance affects utilization

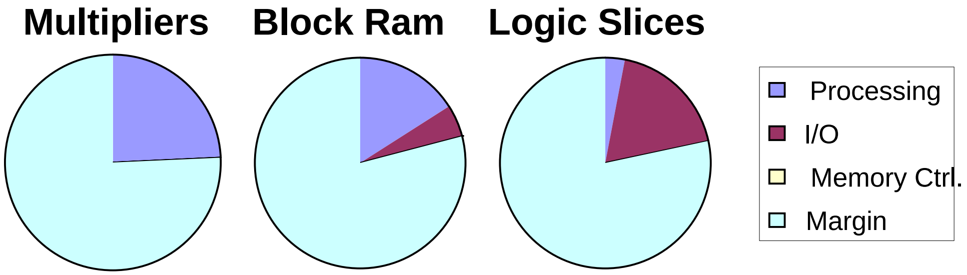


Note:adding additional non-MGT connections could potentially increase throughput

4 Input Module Realization

- I/O and compute bounds are not close -- low utilization
- 36 x 96 unconstrained matrix multiply
- 35 modules required for FPGA digital processor
  - Front-end – 5 modules
  - Small-array beamformer – 24 modules
  - Digital pulse compression – - 6 modules



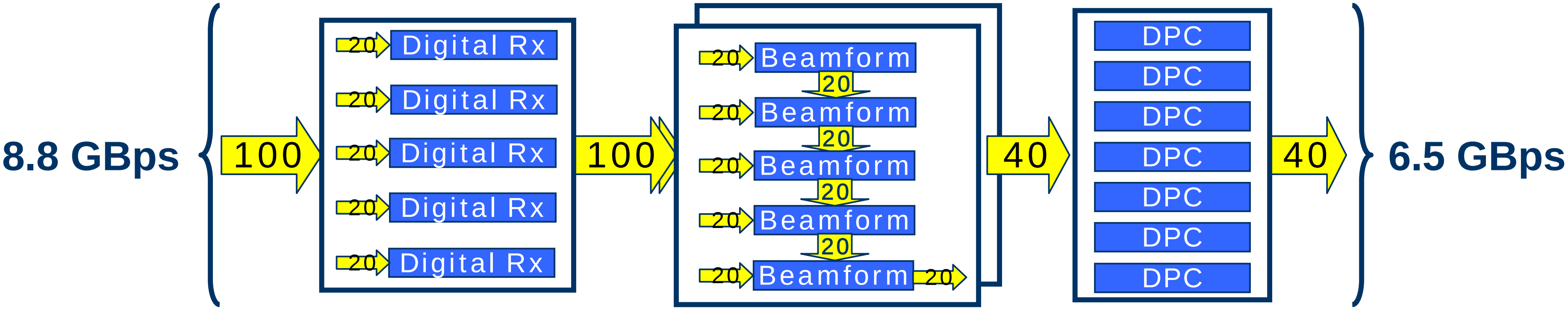


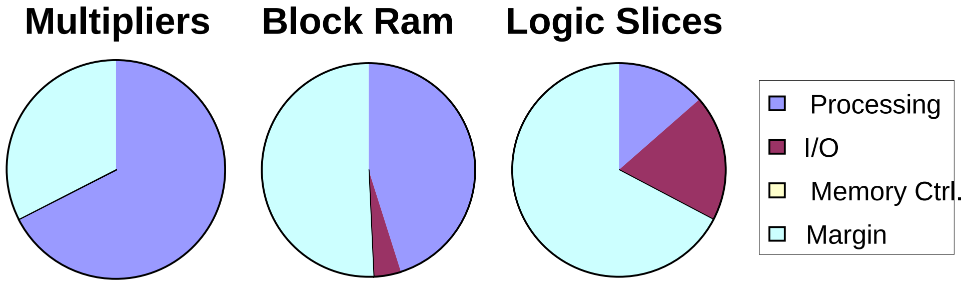
LOW FPGA UTILIZATION

BEAMFORMER MODULE UTILIZATION

20 Input Module Realization

- I/O and compute bounds are close -- good utilization
- 40 x 100 unconstrained matrix multiply
- 22 modules required for FPGA digital processor
  - Front-end – 5 modules
  - Small-array beamformer – 10 modules
  - Digital pulse compression – 7 modules



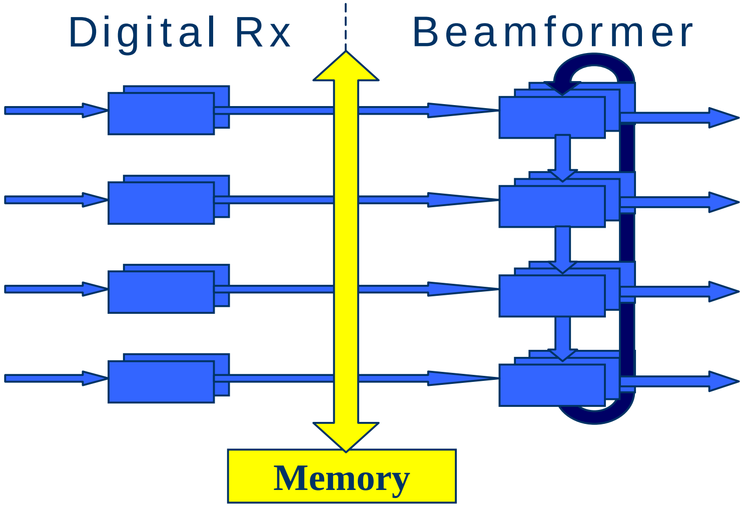


GOOD FPGA UTILIZATION

BEAMFORMER MODULE UTILIZATION

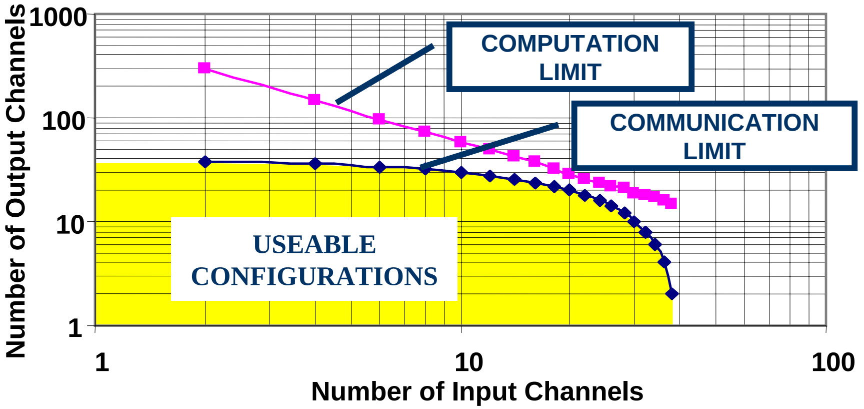
Constrained Linear Architecture

- Use each beamformer module to produce outputs
- MxN constrained complex matrix multiply
  - Use only a subset of inputs for each output
- I/O and computation bounds the as in the unconstrained case
  - Inputs and outputs must be balanced to maximize utilization



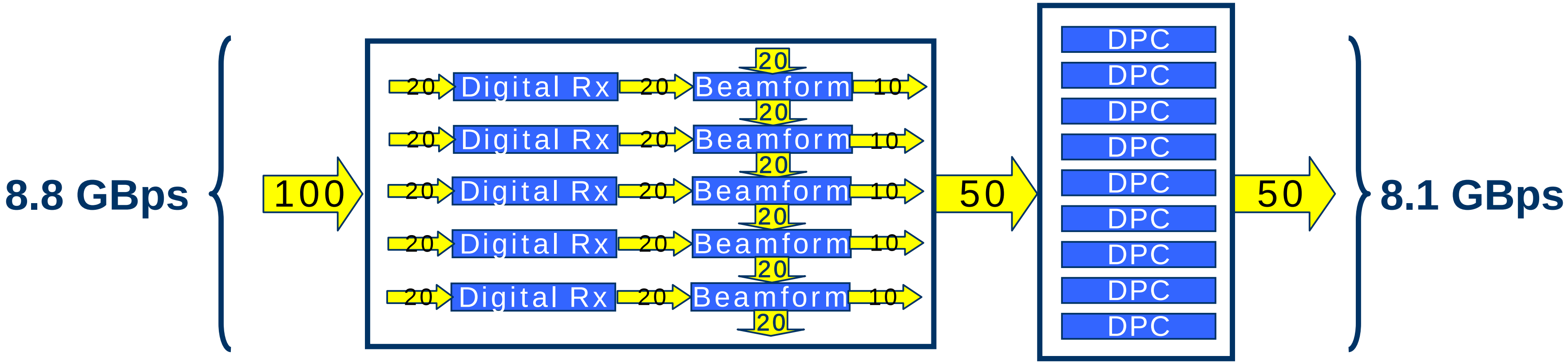
EXPLICIT ZEROS IN BEAMFORMING MATRIX

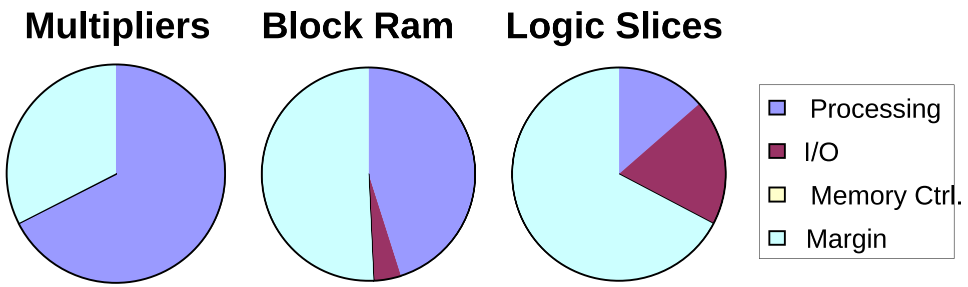
$$\begin{bmatrix} Y_1 \\ \mathrm{M} \\ Y_M \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & \mathrm{K} & 0 & 0 \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} \\ 0 & 0 & \Lambda & H_{(M-1)(N-1)} & H_{(M-1)} \\ H_{M1} & 0 & \Lambda & 0 & H_{MN} \end{bmatrix} \begin{bmatrix} X_1 \\ \mathrm{M} \\ \mathrm{M} \\ \mathrm{M} \\ X_N \end{bmatrix}$$



20 Input Module Implementation

- Adding matrix constraints increases the number of outputs
- 50 x 100 constrained matrix multiply
- 19 modules required for FPGA digital processor
  - Front-end - 5 modules
  - Small-array beamformer – 5 modules
  - Digital pulse compression - 9 modules



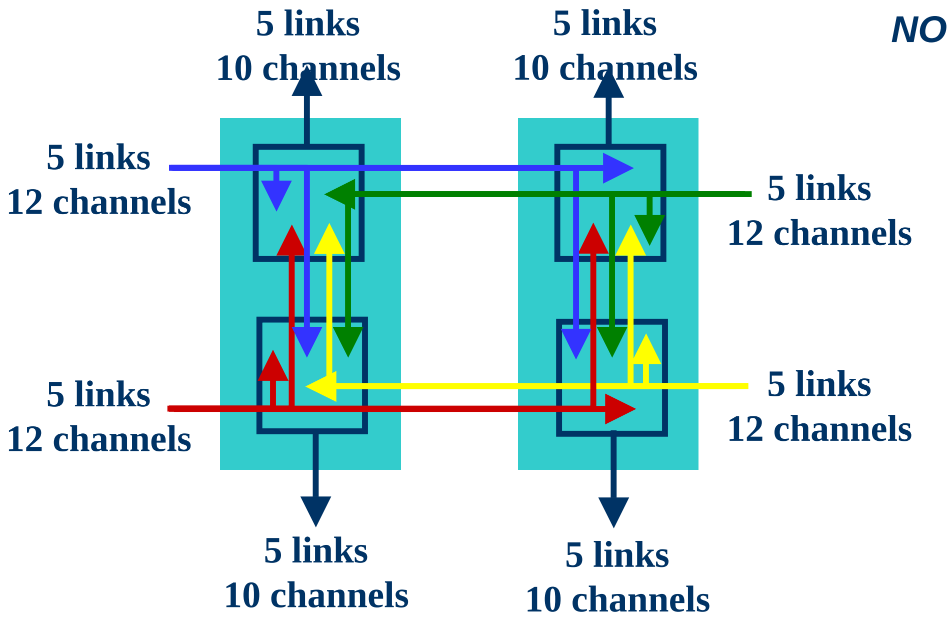


GOOD FPGA UTILIZATION

BEAMFORMER MODULE UTILIZATION

Mesh Architecture

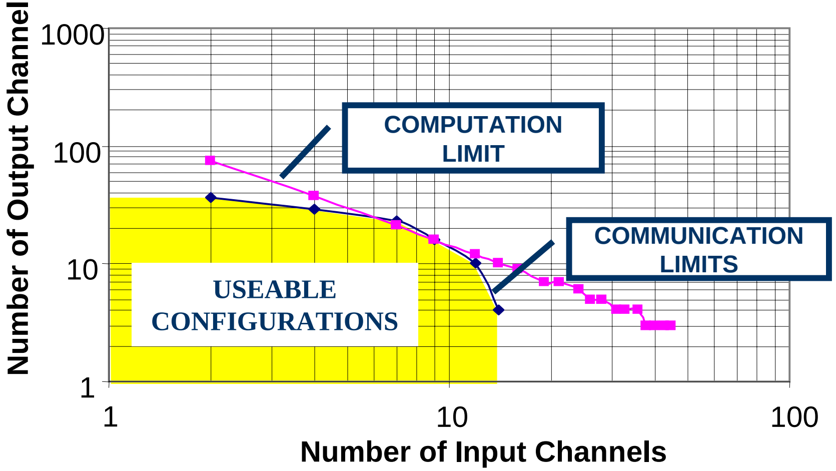
- Mesh architecture offers utilization enhancement
  - I/O and computation bounds touch
- Full unconstrained matrix multiply
- Partially formed beams sent forward for summing in DPC



40 x 48 CMAC using 4 modules

NO EXPLICIT ZEROS IN BEAMFORMING MATRIX

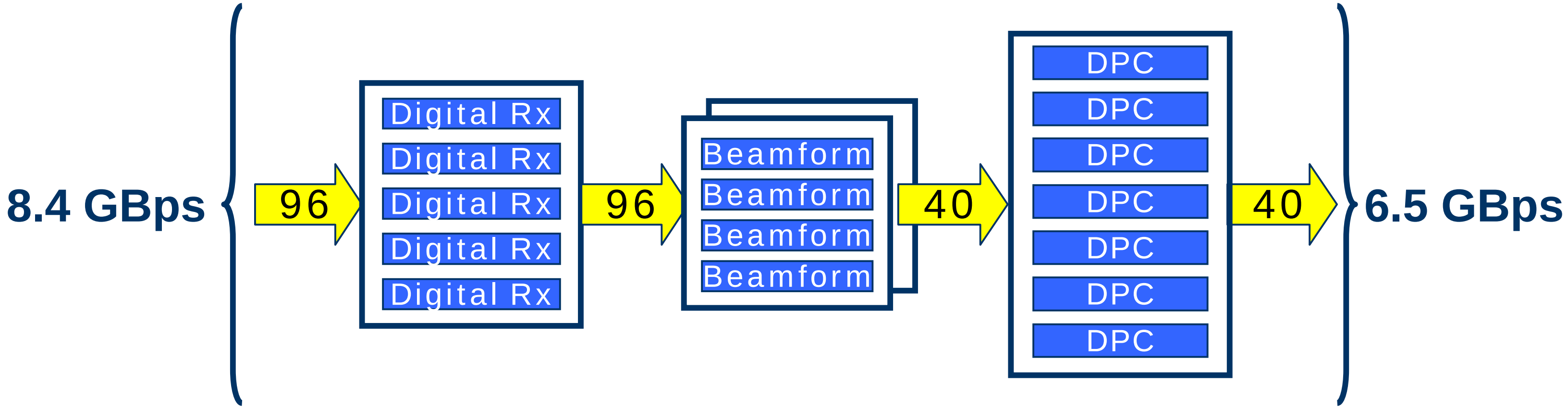
$$\begin{bmatrix} Y_1 \\ \vdots \\ Y_M \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & \cdots & H_{1(N-1)} & H_{1N} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ H_{M1} & H_{M2} & \cdots & H_{M(N-1)} & H_{MN} \end{bmatrix} \begin{bmatrix} X_1 \\ \vdots \\ \vdots \\ \vdots \\ X_N \end{bmatrix}$$

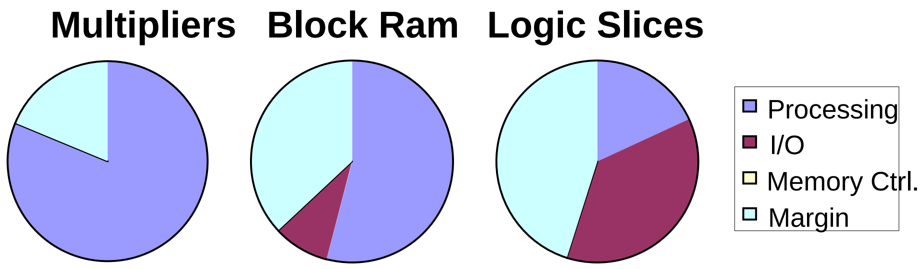


Note: Computation limit normalized for architecture

Mesh Implementation

- I/O and compute bounds touch -- high utilization
- 40 x 96 unconstrained matrix multiply
- 20 modules required for FPGA digital processor
  - Front-end – 5 modules
  - Small-array beamformer – 8 modules
  - Digital pulse compression – 7 modules





BEAMFORMER MODULE UTILIZATION

HIGH FPGA UTILIZATION

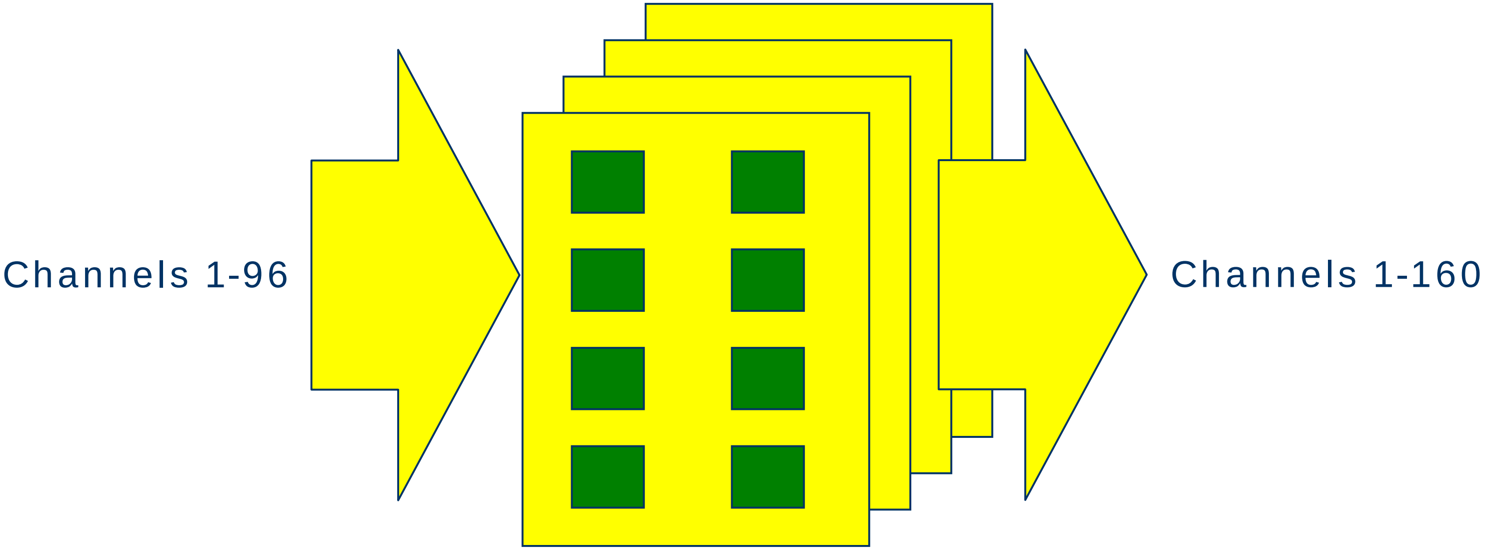
Architecture Comparison

- Mesh architecture gives highest multiplier utilization

	Unconstrained Linear	Unconstrained Linear	Constrained Linear	Mesh
Input Channels	96	100	100	96
Output Channels	36	40	50	40
Beamformer Modules	24	10	5	8
Inputs per Module	4	20	20	12
Multiplies per Output	96	100	40	96
Total Multiplies	3456	4000	2000	3840
Multiplies per Module	144	400	400	480

Large Systems

- Large systems can be created through layering beamformers
  - 8 beam system, 20 channels per beam -- 160 channels
  - 160 x 96 unconstrained matrix multiply
- 65 modules required for FPGA digital processor
  - Front-end - 5 modules
  - Small-array beamformer – 32 modules
  - Digital pulse compression - 28 modules



Summary

Abstract
Presentation
Back to Agenda
Next Abstract

- FPGAs can provide efficient I/O and computational power to address high input bandwidths of modern radar systems.
  - Front-end processing
  - Sub-array beamformer
  - Digital pulse compression
  - Adaptive beamforming
- System topologies that provide efficient utilization of computational and I/O resources change dramatically as system requirements scale.
  - Watch I/O and computation bounds
- Small changes in system requirements can dramatically increase complexity of FPGA implementations when computational bounds of embedded resources is exceeded.
  - Watch for symmetries in filters
  - Watch bit growth before 18-bit multipliers
- FPGAs should be used until application of adaptive beamforming weights due to high bandwidth dataflow.